VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

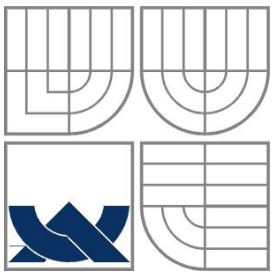
ZABEZPEČENÁ KOMUNIKACE V INTELIGENTNÍ DOMÁCNOSTI

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

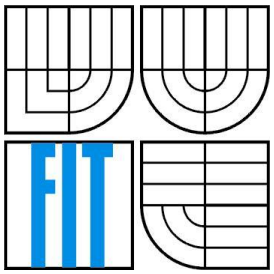
AUTOR PRÁCE
AUTHOR

Petr Brábník

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ZABEZPEČENÁ KOMUNIKACE V INTELIGENTNÍ DOMÁCNOSTI

SECURED COMMUNIKATION IN THE SMART HOME

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Petr Brábník
AUTHOR

VEDOUCÍ PRÁCE — Ing. Tomáš Novotný
SUPERVISOR

BRNO 2014

Abstrakt

Tato bakalářská práce se zabývá tunelováním síťové komunikace mezi adaptérem a cloudem v rámci inteligentní domácnosti. V první části je teoretický rozbor inteligentní domácnosti a síťového tunelování. V další části je vybrán SW pro tvorbu síťových tunelů a jeho následná implementace. Poslední část se zabývá testováním a případným rozšířením funkcionality. Cílem testování je ověřit, zda komunikační tunel splňuje zadané požadavky a v jakém čase je schopen komunikovat.

Abstract

This bachelor thesis deals with tunneling of the network communication between the adapter and the cloud in the area of smart homes. The first part is a theoretical analysis of smart homes and network tunneling. The software for creating network tunnels is chosen and implemented in the second part. The last part deals with testing and possible functionality extension. The goal of testing is to verify that the communication tunnel satisfies specified requirements and how fast can be communication established.

Klíčová slova

OpenVPN, zabezpečení, inteligentní domácnost, šifrování, tunelování síťové komunikace.

Keywords

OpenVPN, security, smart home, encryption, tunneling of network communication.

Citace

Brábník Petr: Zabezpečená komunikace v inteligentní domácnosti, bakalářská práce, Brno, FIT VUT v Brně, 2014

Zabezpečená komunikace v inteligentní domácnosti

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Tomáše Novotného.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Petr Brábník
21. května 2014

Poděkování

Tímto bych chtěl poděkovat vedoucímu mé práce, Ing. Tomáši Novotnému, za jeho ochotu a věnovaný čas. Mé poděkování patří také Ing. Pavlovi Korčekovi za jeho konzultace.

Obsah

1 Úvod

V dnešní době, kdy firmy mohou mít několik poboček a jejich zaměstnanci mohou pracovat z domova, je zapotřebí jedna rozsáhlá firemní síť. Ale vytvářet jednu fyzickou síť je finančně i prakticky nemožné, proto je zapotřebí vytvořit virtuální síť. Pro přístup do této sítě je zapotřebí určitá bezpečnost dat, která se posílají přes více fyzických sítí do této konkrétní virtuální sítě. Asi nejjednodušším řešením tohoto problému je vytvářet VPN (virtuální privátní síť). Tyto VPN vytvářejí síťové komunikační tunely skrze stávající internetovou síť a oddělují přenášená data od ostatních tím, že je šifrují. Takto zašifrovaná data je sice snadné zachytit, ale díky šifrování je téměř nemožné je přečíst.

Práce se zabývá tvorbou síťového tunelu mezi adaptérem a cloudem. Tyto dvě zařízení jsou součástí rozsáhlejšího projektu inteligentní domácnosti. Hlavním parametrem tohoto síťového tunelu je zabezpečení přenášených dat. V následujících kapitolách je popsána teorie inteligentní domácnosti a síťových komunikačních tunelů. Dále jsou zkoumány již existující programy, které problematiku síťových komunikačních tunelů dokáží vyřešit. V části návrhu je vybráno jedno z existujících řešení a je popsáno, jakým způsobem řeší síťové komunikační tunely. Dále je navrženo testování tohoto řešení na předem specifikované požadavky. V kapitole implementace jsou ukázky konkrétní implementace síťového komunikačního tunelu a testovacího programu. V předposlední kapitole je shrnuto testování a naměřené výsledky jsou podrobně rozebrány. Na základě výsledků jsou určeny změny, které by byly zapotřebí pro lepší funkčnost komunikačního tunelu.

2 Teorie

V této kapitole je popsán trend inteligentních domácností, jejich vlastnosti a využití. Je zde rozebrán přístup skupiny zabývající se inteligentní domácností na FIT VUT v Brně. Jsou zde probrány veškeré potřebné znalosti s tímto tématem spojené a hlavně s pojmem síťové tunelování. V poslední části kapitoly jsou popsány již existující řešení síťového tunelování.

2.1 Inteligentní domácnost

Inteligentní domácnosti se na první pohled výrazně neliší od běžných domácností, ale umožňují spojit ovládání vytápění, osvětlení, zabezpečovací systémy, zavlažování, apod. do jednoho centrálního systému. Ten je možno ovládat pomocí jediného nástroje, jako je tablet, chytrý telefon, notebook a mnoho dalších, z různých míst. Tento systém je možno ovládat i mimo prostředí domácnosti a to pomocí internetového připojení. Smyslem inteligentní domácnosti je zvýšit komfort a pohodlí obyvatel. Vstupní investice do tohoto systému by se měla vrátit v podobě úspor energií, protože inteligentní domácnost dokáže například automaticky regulovat vytápění, osvětlení nebo klimatizaci v závislosti na venkovních podmínkách a pohybu osob v jednotlivých místnostech.

Rozvody a zapojení inteligentní domácnosti

Trendem v tvorbě inteligentních domácností je vytvořit kabelové rozvody již při výstavbě nového domu nebo při rekonstrukci starého. Celá tvorba rozvodů závisí na zpracovaném projektu inteligentní domácnosti. V projektu se může počítat i s prvky, které nebudou ihned k dispozici, ale mohou se instalovat až později. Proto se pro tyto prvky vytvoří přípojka již při výstavbě, protože pozdější tvorba přípojky sekáním do zdi popřípadě maskováním přívodního kabelu není dobré řešení.

Bezdrátové rozvody jsou další možností instalace inteligentní domácnosti. Bezdrátové technologie mají nespornou výhodu v tom, že se dají instalovat bez většího stavebního zásahu. U těchto technologií ale mohou nastávat problémy se spolehlivostí. Musí se řešit vzájemné rušení prvků a celková komunikace mezi prvky a přijímačem. Dále mají tyto technologie omezený dosah a pro zvětšení dosahu se musí instalovat dodatečná zařízení, která ho rozšíří. [1]

Topení

Topení v inteligentní domácnosti zpříjemňuje bydlení a snižuje provozní náklady na vytápění. Jelikož termostat není v jedné místnosti, ale je v každé místnosti, je možné si pro každou místnost nastavit jinou teplotu. Proto je tedy možné v obývaných pokojích nastavit příjemnou teplotu, naopak v chodbě či ložnici můžeme teplotu snížit. Na každém topení je nainstalována digitální termostatická hlavice, která je připojena k inteligentní domácnosti a řídí teplotu každého topení zvlášť. Změna teploty se

provádí přes uživatelské rozhraní inteligentní domácnosti stejně jako ovládání ostatních prvků. K úspoře prostředků dochází, pokud inteligentní domácnost vyhodnotí, že nikdo není doma a sníží teplotu vytápění všech místností. Dále je možné si nastavit, v kterých časech se členové domácnosti vrací domů a nějaký čas před tím zvýšit teplotu v domě. Díky tomu přichází člen rodiny do vytopeného domu a nemusí čekat, než se dům vytopí poté, co ručně pustil topení.

Stejně tak jako topení se může chovat i klimatizace v teplých měsících. Také klimatizace může být spouštěna podle daného časového plánu nebo pomocí pohybových čidel v místnostech. Pokud čidlo zaznamená pohyb, přepne klimatizaci z úsporného režimu do pracovního režimu a ochladí místnost na požadovanou teplotu. Rozdíl mezi teplotou v úsporném režimu a pracovním režimu může být několik stupňů, čímž se šetří náklady spojené s klimatizací, protože není zapotřebí ochlazovat místnost na tak nízkou teplotu, když v ní nikdo není. [2]

Osvětlení

Osvětlení je možné ovládat stejně jako topení vzdáleně přes uživatelské rozhraní například v chytrém telefonu nebo tabletu. V inteligentní domácnosti je možné nastavovat různé scény osvětlení pro různé příležitosti. Například pro sledování filmů se nastaví scéna s mírným osvětlením za divákem pro lepší požitek ze sledování filmů. Tyto scény je možné nastavit pro mnoho dalších příležitostí. Úspora v podobě svícení v inteligentní domácnosti přichází s možností regulovat intenzitu svitu jednotlivých světelných zdrojů. Dále je možné používat v místnostech čidla pohybu nebo tepelná čidla, čímž se dá určit, jestli v místnosti někdo je, a pokud není, osvětlení se automaticky vypne a tím šetří náklady na energii. Dále se tyto čidla dají využívat pro účely rozsvícení světla v noci na chodbě, když jde člověk na toaletu. [3]

Bezpečnost

Systém inteligentní domácnosti myslí i na bezpečnost obyvatel a majetku. Hlavním pilířem je centrální kamerový systém, který zaznamenává dění v okolí nemovitosti. Podle potřeb klienta může být kamerový systém nainstalován i uvnitř nemovitosti, samozřejmě podle představ klienta, aby bylo zajištěno jeho soukromí. Veškeré záznamy jsou ukládány na bezpečné úložiště, aby mohly být v případě potřeby přehrány nebo je možnost tyto záznamy vypálit na DVD či jiné přenosné médium. Kamerový systém nemusí hlídat jen případné nežádoucí osoby, ale například i děti hrající si u bazénu. Díky tomu je možné na jakémkoli zařízení, které umožňuje připojení k inteligentní domácnosti, hlídat děti, aniž by rodič musel být fyzicky s nimi u bazénu.

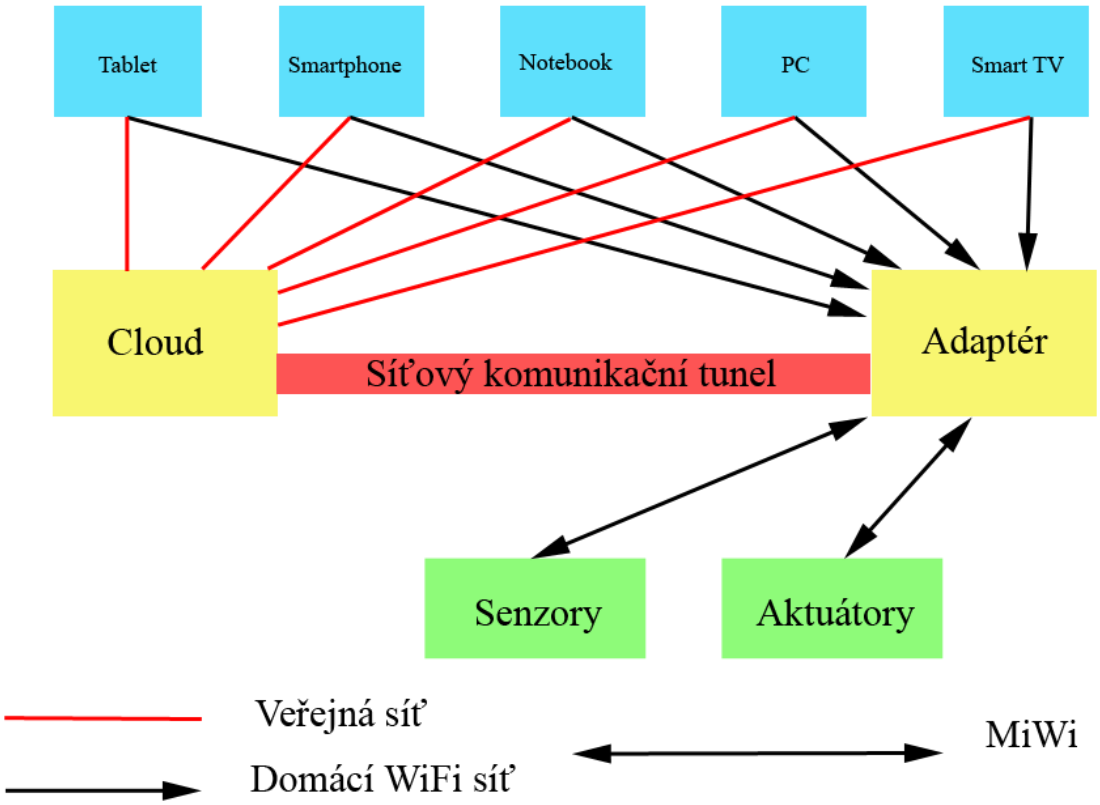
Dalším prvkem bezpečnosti je systém odemykání a uzamykání domácnosti. To je možné řešit pomocí čipů podobných těm na platebních kartách. Tyto čipy mohou být zabudovány do přívěšků na klíče, náramků a mnoho dalších. Dále je možné objekt uzamknout nebo odemknout pomocí aplikace v chytrém telefonu či tabletu. Zvonek u dveří nebo vchodové branky může být napojený na

inteligentní domácnost. Pokud má zvonek zabudovanou kameru, je možné si na tabletu či telefonu zobrazit obraz a tím zjistit, kdo je u dveří. Jestliže není nikdo doma, může inteligentní domácnost upozornit klienta na mobilní telefon a vytvořit hovor mezi jeho telefonem a zvonkem, buď jako videokonferenci, nebo jako klasický hovor. [4]

2.2 Projekt inteligentní domácnosti na FIT VUT v Brně

Na Fakultě informačních technologií VUT v Brně (FIT) vznikla skupina zabývající se komplexním technickým vývojem inteligentní domácnosti. Na obrázku č. 1 je vidět návrh architektury pro inteligentní domácnost. Je zde také vidět, že celá architektura je rozdělena do několika vrstev, které různými způsoby vzájemně komunikují. Na nejnižší úrovni jsou senzory a aktuátory, které komunikují s adaptérem pomocí bezdrátové technologie. Předpokládá se použití technologie MiWi. Nabízelo by se využití WiFi technologií, ale její použití je drahé, složité a energeticky náročné. Na další vrstvě je cloud a adaptér. Ty komunikují pomocí síťového tunelu. Na nejvyšší vrstvě jsou zařízení, pomocí kterých uživatel ovládá inteligentní domácnost (tablet, smartphone, notebook, atd.). [5]

Hlavní myšlenkou tohoto návrhu je používat již existující prvky, které se pouze připojí k adaptéru a tím rozšíří funkčnost inteligentní domácnosti. To znamená, že každý bude moci inteligentní domácnost přizpůsobit svým potřebám a finančním možnostem a postupně jí rozšiřovat připojováním nových prvků.



Obrázek 1: Návrh architektury [5].

Cloud

Cloud je server, který neposkytuje pouze služby a datové úložiště, ale obsahuje i logiku aplikací inteligentní domácnosti. Může poskytovat služby v závislosti na příchozích datech z koncových prvků (například předpověď počasí na základě tlaku, vlhkosti ovzduší, rychlosti větru, teploty). Na cloud by se registrovaly jednotlivé adaptéry a zasílaly by se informace od koncových prvků ke zpracování. Cloud by s daty nakládal podle vnitřně předdefinované logiky a adaptér by po jejich vyhodnocení zasílal příkazy pro práci s koncovými prvky.

Výhodou tohoto řešení pro uživatele je to, že z internetu by se mohl dostat do inteligentní domácnosti přes tento server například přihlášením na webové rozhranní cloudu, aniž by si museli pamatovat IP adresu cloudu (což je v případě IPv6 nepřípustné), ale jen jeho doménové jméno. Měl by vygenerovanou dvojici uživatelského jména a hesla, kterou by prokázal svou identitu vůči cloudu.

Nevýhodou tohoto řešení je, že veškerá uživatelská data jsou uložena vzdáleně na cloudu, proto se musí řešit jejich zabezpečený přenos a autorizace přístupu k nim. Dále v případě výpadku připojení jsou tato data nedostupná.

Adaptér

Adaptér je v našem případě vývojová platforma od firmy Olimex s nainstalovaným systémem Linux, která dělá prostředníka mezi cloudem a koncovými prvky. Zpracovává data ze senzorů a přeposílá tyto informace na cloud. Dále pak data z cloudu předává do aktuátorů, které na tyto informace zareagují a provedou požadovanou činnost. Adaptér musí mít jedinečnou identifikaci, díky které je schopný zaregistrovat se do cloudu.

Adaptér musí umět komunikovat se všemi zařízeními, které jsou k němu zaregistrované, a to i na různých komunikačních protokolech. Oproti koncovým prvkům musí být adaptér kdykoliv k dispozici, takže u něho nedochází k uspávání.

Propojení adaptéru k domácímu routeru, který tvoří bránu pro přístup na internet, je možné provést několika způsoby. Může být vestavěný přímo v routeru, ale to znamená zásah jak do jeho HW, tak i do SW, což není příliš jednoduchá varianta. Lepší variantou je připojení k USB routeru, pokud to rozhraní nabízí. U této varianty je zapotřebí zásah do SW (nahrání ovladačů adaptéru), aby spolu router a adaptér uměly komunikovat. Jako další varianta se nabízí adaptér připojit pomocí WiFi, nebo Ethernetu. Ani jedna z těchto technologií nevyžaduje zásah do SW či do HW routeru, což se jeví jako velká výhoda. U těchto variant je ale zapotřebí vyřešit napájení adaptéru, protože není součástí routeru. Řešením je napájení buď pomocí Ethernetu (PoE), nebo přímým připojením do elektrické sítě. U technologie WiFi je omezení v tom, že je potřeba nejprve do adaptéru zadat SSID sítě a případné heslo, aby se adaptér mohl připojit.

Adaptér nemusí tvořit pouze prostředníka mezi cloudem a koncovými prvky, ale může obsluhovat i některé základní funkce. Například při výpadku internetového připojení by byla inteligentní domácnost nepoužitelná, proto musí adaptér umět obsloužit základní komponenty i bez připojení ke cloudu.

Senzory a aktuátory

Senzory jsou prvky, které zaznamenávají konkrétní veličiny (měření teploty, vlhkosti, slunečního svitu apod.) a předávají je adaptéru. Oproti tomu aktuátory jsou prvky, které vykonávají jednu konkrétní činnost (rozsvícení světla, zatažení/roztažení rolet, sepnutí topení apod.) podle pokynů od adaptéru. Oba tyto prvky by měly být jednoduché, převážně jednoúčelové a zároveň co nejlevnější. Zařízení budou s adaptérem komunikovat protokolem MiWi [6]. MiWi je jednoduchý bezdrátový protokol. Je navržený pro bezdrátovou komunikaci na krátkou vzdálenost (do 50 metrů). Je energeticky málo náročný, což je pro tato zařízení jeden z hlavních parametrů.

Každý prvek se musí umět jednoznačně přihlásit k adaptéru. Toto přihlášení musí být uživatelsky co nejsnadnější. Například pomocí stisku tlačítka na koncovém zařízení a současným stiskem tlačítka na zařízení pro příjem MiWi signálu, který musí být připojen k adaptéru. Protože přihlášení musí být jednoznačné, musí mít každý prvek jedinečné označení (např. MAC adresu).

Zařízení dále musí umět předat adaptéru základní informace o sobě, aby si adaptér tyto informace mohl uložit.

Pro napájení koncových zařízení se předpokládá použití baterií, pokud není možné toto zařízení připojit přímo do elektrické sítě. Z elektrické sítě by mohly být napájeny například opakovače signálu, které jednoduše zopakují příchozí signál a tím rozšíří dosah signálu pro vzdálenější zařízení. Baterie musí mít dlouhou životnost a v tomto případě mohou být vestavěné. Pokud by baterie nevydržely dostatečně dlouho, musí mít uživatel možnost tyto baterie vyměnit. O stavu baterií musí systém uživatele informovat, aby mohl včasně zareagovat na jejich slábnoucí kapacitu a vyměnit je. Výměna baterií by měla být co nejsnadnější (např. bez použití nářadí). Kvůli výdrži baterie musí zařízení umět omezit svůj provoz na určitý časový interval. Senzory mohou na adaptér zasílat data jednou za předem definovaný interval, proto se mohou uspat na dobu do příštího vysílání a tím šetřit spotřebu energie baterií. Ne všechny aktuátory musí být v nepřetržitém spojení s adaptérem, některé mohou svou činnost vykonávat s drobným časovým odstupem. To znamená, že se mohou na nějaký krátký časový interval uspat a při probuzení se adaptéru dotázat, jestli mají vykonat konkrétní činnost.

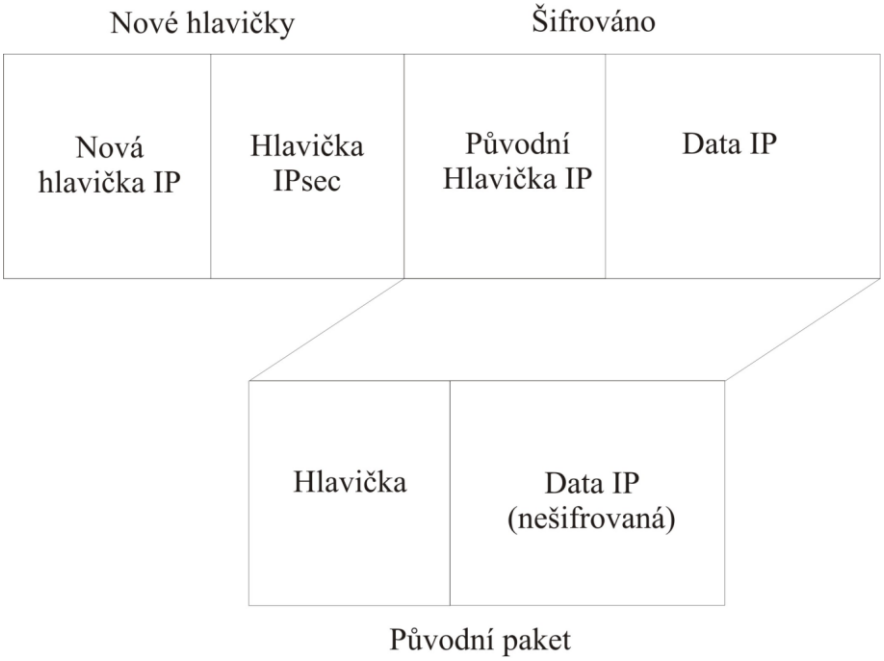
2.3 Tunelování síťového provozu

Tunelování je proces zapouzdření jednoho typu paketu do jiného tak, aby byl možný vhodný transport. V podstatě to znamená, že systém vezme celý paket a vloží jej do jiného paketu, který jej zastřešuje. Tento paket se následně přenáší po síti a u cílového systému se zapouzdřený paket vyjme a tím opustí prostor tunelu. Tunelování umožňuje přenášet data přes nekompatibilní sítě, obcházet některá administrativní omezení sítě, a pokud je tak nastaveno, tak zabezpečovat komunikaci přes nezabezpečenou síť.

Do tunelování jsou zapojeny tři různé protokoly:

1. původní datový protokol
2. protokol, kterým se zapouzdřuje původní protokol
3. protokol stávající sítě

Do zapouzdřeného paketu se musí doplnit hlavička, které bude rozumět síť, přes kterou vede tunel (obvykle IP). To lze použít například pokud síť podporuje jen IPv4, ale cílem je komunikace pomocí protokolu IPv6 nebo i jiných méně používaných protokolů jako je IPX/SPX, AppleTalk apod. Tunelování samo o sobě nezajišťuje, že data budou bezpečně přenášena. Pro zabezpečené tunelování je nutné data šifrovat například protokolem IPsec.



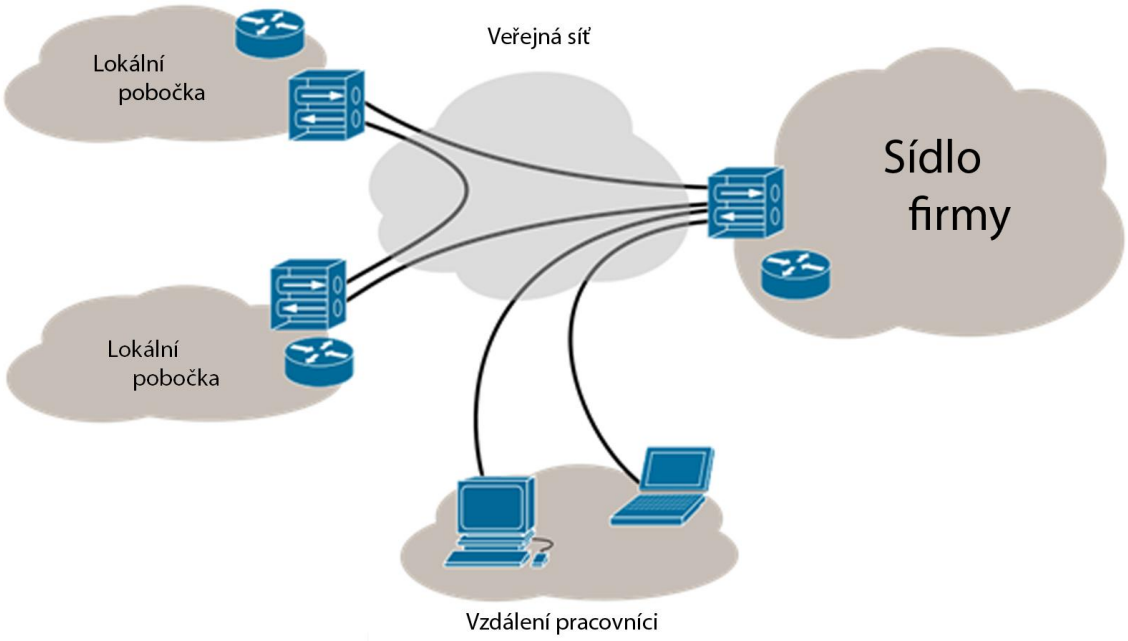
Obrázek 2: Zapouzdření původního paketu [7].

Nevýhodou tunelování je, že na obou stranách tunelu je potřeba zpracovat pakety, což zvyšuje vytížení procesoru a omezuje rychlost přenosového pásma. U tunelů typu klient-klient je tento nárůst zanedbatelný. Ale u rozsáhlých sítí, kde se pomocí tunelu připojují k jednomu cílovému serveru stovky až tisíce klientů, znamená zpracování paketů velký nárůst vytížení procesoru. V případě šifrování se tyto nároky několikanásobně zvyšují, a proto se pro jejich snížení používají hardwarové kryptografické akcelerátory. Tyto komponenty ale celkově zvyšují cenu vybudování tunelu. [7]

2.4 Virtuální privátní sítě

Před příchodem sítí VPN využívaly firmy k propojení poboček a hlavní firemní sítě drahé privátní připojení, realizované pomocí pronajímané linky nebo okruhu frame relay. Hlavním přínosem zavedení VPN je používání stávající veřejné internetové sítě. Pobočky se připojují k firemní síti z různých míst na světě, ale přesto se díky VPN tváří jako by k ní byly fyzicky připojeny. Použití VPN pro malé firmy (desítky až stovky zaměstnanců) je finančně nenáročné a existují i implementace, které jsou zdarma, což firmám šetří velké množství prostředků. Pro větší firmy je potřeba počítat s navýšením nákladů na vybudování VPN sítě. Tyto firmy musí nakoupit specializované hardwarové prvky, které ovšem zlepší parametry přenosu pro tyto sítě. Náklady spojené s nákupem těchto prvků jsou ale v časovém měřítku několika měsíců až let menší, než náklady za pronájem privátní linky. Ale nejen úspora prostředků je přínosem VPN sítí. Dále pak VPN sítě zvyšují flexibilitu pracovníků. Umožňují jim přístup na firemní síť pomocí jakéhokoliv připojení

k internetu, jako je mobilní připojení, veřejné WiFi, domácí připojení k internetu a to vše, aniž by se museli bát o bezpečnost přenášených dat. [8]



Obrázek 3: Příklad firemní VPN sítě [9].

Šifrování

Základním konceptem VPN technologie je zabezpečení komunikačního kanálu pomocí šifrování. Komunikační kanál může být šifrován na odlišných vrstvách ISO/OSI modelu.

1. Na aplikační vrstvě může být šifrování zajištěno programově například pomocí zabezpečeného kanálu SSH (Secure Shell).
2. Na transportní vrstvě lze použít protokol pro ochranu obsahu komunikace mezi dvěma stranami jako je například Secure Sockets Layer (SSL). Běžně se tento způsob používá pro komunikaci s webovým prohlížečem.
3. Na síťové vrstvě se používá převážně protokol IPsec, který nešifruje pouze obsah paketu, ale i TCP/IP informace. Jediné informace, které jsou ke zjištění případným útočníkem, jsou IP adresy zdroje a cíle komunikace, které jsou zapotřebí ke směrování paketu. Jinak veškeré ostatní informace jsou šifrované. [10]

IPsec přidává bezpečnostní mechanizmus do standardní IP vrstvy, tudíž je nezávislý na protokolech vyšších vrstev. IPsec definuje dva bezpečnostní mechanizmy. [11]

Prvním z nich je autentifikace, která zaručuje původ dat. Příjemce si může ověřit, že přijatý IP paket opravdu patří tomu, od koho byl paket přijat. Protokolem, který zajišťuje autentifikaci, je Authentication Header (AH).

Druhý mechanismus je kryptování. Všechno kromě IP hlavičky je zašifrováno pomocí předem domluveného algoritmu. Příjemce musí umět tento paket rozšifrovat. To znamená, že před samotnou komunikací se musí obě strany domluvit na způsobu šifrování. Protokolem, který zajišťuje šifrování je Encapsulating Security Payload (ESP). [12]

2.5 Zkoumání existujících nástrojů pro zabezpečený přenos informací

V této kapitole jsou zkoumány existující nástroje pro zabezpečený přenos informací mezi adaptérem a serverem.

OpenVPN

OpenVPN je volně dostupný program pro vytváření VPN tunelů. Jeho použití je velice jednoduché a intuitivní. Je k němu k dispozici spousta návodů a má rozsáhlé manuálové stránky. OpenVPN nepoužívá IPsec, ale SSL/TLS, což umožňuje použití certifikátů X.509 pro zabezpečení komunikace. Pro každý směr komunikace umožňuje použít jiný klíč. Také umožňuje nastavit velikost replay-okna, což zvyšuje bezpečnost, protože je snížena možnost prolomení pomocí opětovného přehrání posílaných dat. Dále je možné použít jakýkoliv šifrovací algoritmus z knihovny SSL.

OpenVPN umožňuje komunikaci klient-klient, ale i komunikaci klient-server. Jako výchozí OpenVPN používá UDP protokol, ale přepsáním jednoho řádku v konfiguračním souboru umožní používání TCP protokolu. Veškeré nastavení OpenVPN probíhá v jednom konfiguračním souboru tím, že se připisují na jednotlivé řádky příkazy zastupující požadovanou funkčnost. Veškerá komunikace probíhá na jednom portu, proto je snadné nastavit případný firewall. Mimo jiné OpenVPN umožňuje kompresi dat a zasílání tzv. ping-of-live, který se v pravidelných intervalech dotazuje na dostupnost tunelu a tím ho udržuje spuštěný. OpenVPN umožňuje nastavení automatického spuštění klienta při spuštění systému a automatického navázání tunelu. [13]

Při použití OpenVPN tunelu je nárůst zpoždění paketu zhruba 25% oproti běžnému provozu. Toto zpoždění je způsobeno režií spojenou s posíláním dat skrze OpenVPN tunel. Režie je zapříčiněna především zabalením paketu na straně odesílatele a následným rozbalení na straně příjemce.

SSH

U SSH se tunely vytváří tzv. přeposíláním portů (port forwarding). Spojení SSH náhodně vybere lokální port a přiřadí jej určitému vzdálenému hostiteli a vzdálenému portu. Po navázání spojení klient vyčkává na příchozí komunikaci na vybraném portu a veškerý provoz směrovaný na tento port odešle do vzdáleného SSH serveru. Tunel je vždy vytvořen pro konkrétní vzdálený port, a proto je potřeba samostatný tunel pro každou dvojici hostitel a port, nad kterou je potřeba přenášet data. Přes SSH tunely lze posílat pouze pakety protokolu TCP, pakety ostatních protokolů jako je UDP přes SSH tunel posílat nelze. Ve většině unixových operačních systémů je SSH klient součástí výchozí instalace. Pokud není součástí výchozí instalace, je snadné ho zdarma stáhnout a nainstalovat (například OpenSSH). Na systému Windows v základní instalaci SSH klient chybí a proto je nutné ho vždy stáhnout manuálně (například PuTTY). Serverový software tzv. démon bývá také součástí výchozí instalace u unixových systémů (obvykle sshd).

CiscoVPN

Pro vytvoření CicsoVPN je potřeba pořídit specializovaný router nebo specializovaný prvek, který vytváří VPN server. Toto zařízení výrazně navyšuje náklady na vytvoření tohoto síťového komunikačního tunelu. CiscoVPN klient je volně ke stažení na internetových stránkách Cisco a snadno instalovatelný na všechny běžně používané operační systémy. Pro instalaci CiscoVPN serveru je zapotřebí znát programovací jazyk, který je používán na Cisco zařízeních. Cisco zařízení mají vlastní operační systém, který se nazývá Cisco IOS. IOS příkazový řádek obsahuje pevnou množinu prvků podle toho, v jakém módu se programátor právě nachází. IOS obsahuje čtyři módy a to uživatelský mód, privilegovaný mód, mód konfigurace rozhraní a konfigurační mód. Díky těmto vlastnostem není instalace CiscoVPN triviálním úkonem. Je zapotřebí mít určité základní znalosti o programování na Cisco zařízeních. Dále je zapotřebí vědět, jak bude konečný VPN tunel vypadat, protože u CiscoVPN je nutné nastavit kompletní funkčnost. Oproti předchozím řešením je toto výrazně složitější. U OpenVPN a SSH je možné zprovoznit jednoduchý tunel se slabou funkčností a postupně funkčnost rozšiřovat.

3 Návrh

V této kapitole jsou specifikovány požadavky na síťový komunikační tunel. Na základě specifikovaných požadavků je vybráno jedno z možných řešení, které jsou zmíněny v kapitole 2.5. V této kapitole je také navrženo testování, jehož hlavním cílem je simulovat reálné problémy, které mohou v domácnosti nastat. Hlavním měřeným faktorem testování je čas obnovy spojení mezi adaptérem a cloudem.

3.1 Požadavky na síťový komunikační tunel

Pro síťový komunikační tunel byly stanoveny, v rámci této práce a Inteligentní domácnosti na FIT VUT v Brně, tyto požadavky:

1. *Oboustranný šifrovaný tunel pro zasílání dat a příkazů.* Tunel musí být zabezpečen například pomocí certifikace a musí pomocí něj být možné posílat data a příkazy z adaptéru na cloud a obráceně.
2. *Komunikace bez pollingu.* Zahájení komunikace musí být možné ze strany adaptéru i ze strany cloudu bez dotazování na funkčnost komunikačního síťového tunelu.
3. *Nízká latence.* Odezva mezi cloudem a adaptérem musí být co nejnižší (v řádech milisekund).
4. *Izolace jednotlivých klientů.* Každý klient musí vytvářet vlastní tunel. Každý tunel musí být nezávislý vůči ostatním klientům.
5. *Správa klíčů.* Výměna klíčů mezi cloudem a adaptérem nesmí být příliš náročná. Ať už časově nebo nároky na výpočetní sílu cloudu/adaptéru. Předávání klíče musí být bezpečné vůči odposlechnutí, podvržení a jiným útokům.
6. *Nízká náročnost na systém.* Adaptér bude vestavěný systém s omezenými paměťovými i výpočetními možnostmi, tudíž celý tunel musí být co nejjednodušší.
7. *Nezávislost na grafickém prostředí.* Na vestavěném systému nebude grafické rozhraní, tudíž veškerá obsluha musí probíhat v terminálu.
8. *Rychlá reakce na ztrátu síťového spojení a následná signalizace napojeným klientům.* V případě ztráty síťového spojení musí být adaptér schopen spojení co nejrychleji obnovit.
9. *Potenciál v obsluze velkého množství klientů.* Cloud musí zvládnout připojení většího množství klientů, aniž by se prodloužila doba jejich obsluhy. Ve výsledku se ke cloudu mohou připojovat až tisíce uživatelů.
10. *Možnost rozšíření o load-balancing a failover.* Musí umožnit tvorbu záložních serverů. Proto při výpadku primárního serveru je k dispozici sekundární server, který převezme obsluhu adaptérů. Dále musí umožnit rozdělení zátěže mezi více serverů jako je tomu například u DNS, kde jsou servery rozděleny do zón a každý primární server má vlastní sekundární server.

11. *Funkčnost v běžném síťovém prostředí.* Jelikož tunel vytváří adaptér, musí si poradit s průchodem například přes NAT, protože každý uživatel doma nemá veřejnou IP adresu.

3.2 OpenVPN

Ze zmíněných řešení v kapitole 2.5 této práce je vybráno řešení pomocí OpenVPN. Toto řešení nejlépe odpovídá požadavkům z kapitoly 3.1 této práce. Díky tomu, že je OpenVPN volně ke stažení, finančně nezatěžuje celý projekt. Ke zprovoznění síťového komunikačního tunelu postačí rozdistribuovat klíče, stáhnout a nainstalovat OpenVPN software.

Server

Na straně serveru je potřeba nainstalovat stejný SW, jako je na straně klienta. V dalším kroku je potřeba vytvořit konfigurační soubor a získat certifikát. Tento konfigurační soubor se oproti klientskému konfiguračnímu souboru liší a tím určuje, že bude spuštěno OpenVPN v módu serveru. Následně se na serveru vytvoří klíče jak pro server, tak pro klienty. V případě OpenVPN nemusí být server speciální zařízení, postačí stejné zařízení, jako je na straně klienta, jen s jiným konfiguračním souborem. Toto zařízení ale musí být schopné obsloužit všechny klienty a proto je zapotřebí HW, který toto zvládne. Se zvyšujícím se počtem obsluhovaných klientů rostou i nároky na HW vybavení a jeho výkon.

Klient

Po nainstalování potřebného SW na klientské zařízení (adaptér) bude nutné vytvořit konfigurační soubor. Dalším krokem bude získání certifikátů. Ty je zapotřebí přenášet co nejbezpečnější cestou, aby je případný útočník nebyl schopný zachytit a použít. Ideálním řešením je použít přenosné médium jako například flash disk. Nejnebezpečnější varianta je klíče přenášet nezabezpečenou formou přes internet. Pokud jsou certifikáty uloženy na adaptéru, je možné spustit OpenVPN tunel. V případě, že je na straně serveru vše v pořádku a na cestě mezi klientem a serverem nenastal žádný problém, je možné přes vytvořený tunel začít přenášet data a příkazy bezpečně.

Certifikační autorita

Pro zaručení bezpečnosti bude vytvořena certifikační autorita a budou jí podepsány klíče. Certifikační autorita je organizace, která vydává digitální certifikáty. Tím, že vydá certifikát, potvrzuje pravdivost údajů, které jsou ve volně dostupném veřejném klíči uvedeny. Digitální certifikáty se používají při přístupu na webové servery, podepisují se s nimi emaily a jiná data.

3.3 Návrh testování

V této části práce je navrženo testování OpenVPN komunikačního tunelu. Toto testování je navrženo s ohledem na požadavky specifikované v kapitole 3.1 této práce.

Skript

Pro účely testování bude navržen krátký skript. Úkolem tohoto skriptu bude vytvořit požadované množství VPN klientů. Každý klient bude mít vlastní soubor, do kterého se budou ukládat logy.

Bude napsán v jazyce Bash, aby byl spustitelný bez nutnosti překladu. Bash je v informatice název pro unixový shell a vytváří interpret pro příkazový řádek. Jeho hlavní výhodou je, že je možné ho spustit téměř na všech unixových operačních systémech.

Tento skript bude ve smyčce kopírovat jeden výchozí konfigurační soubor. V každém průchodu smyčky se bude zvyšovat čítač. K názvu kopie bude přidávat hodnotu čítače. Následně v tomto souboru změní řádek, ve kterém je název souboru s logy. K názvu přidá hodnotu čítače, aby název konfiguračního souboru odpovídal názvu log souboru. Toto všechno je důležité pro přehlednost následné kontroly, jestli vše proběhlo správně. Zadání počtu vytvářených klientů bude z příkazové řádky. Veškeré informace o průběhu a výsledku skriptu budou vypsány do terminálu.

Skript bude postupně spouštěn s hodnotami od 1 do 1000. Hodnoty se budou zvyšovat po sto připojených klientech a výsledky budou zaznamenávány. Bude se zkoumat zatížení serveru, především vytížení CPU, paměti a odezva síťové karty.

Testování bude prováděno na serveru ant-2, který má tyto HW parametry:

- CPU: čtyřjádrový Intel Xeon Processor E5410
- paměť: 10 GB RAM
- síťová karta: Intel Corporation 80003ES2LAN Gigabit Ethernet Controller

Výstupem tohoto pokusu bude graf, na kterém budou zobrazeny HW nároky v závislosti na počtu připojených klientů.

Reakce na reálné situace, které mohou v domácnosti nastat

Dalším bodem testování bude měření času při konkrétních situacích, které v inteligentní domácnosti mohou nastat. Při těchto testech bude měřena doba, za kterou se obnoví spojení mezi adaptérem a cloudem.

V první fázi bude adaptér odpojen od elektřiny a následně opět připojen. Tímto se bude simulovat výpadek napájení adaptéru. Měření bude spuštěno ve chvíli, kdy bude k adaptéru znovu připojen napájecí kabel, a zastaveno bude ve chvíli, kdy na straně cloudu přijde odpověď na příkaz ping.

Jako další bude adaptér odpojen od síťového připojení. Po opětovném připojení bude měřena doba, za kterou bude schopen opět navázat spojení s cloudem skrze VPN tunel. Tato situace bude simulovat výpadek domácího routeru. Měření bude spuštěno ve chvíli, kdy bude síťový kabel znovu připojen k adaptéru. Měření bude zastaveno ve chvíli, kdy přijde odpověď na příkaz ping od adaptéru na cloudu.

V poslední fázi bude adaptér připojen na domácí router a od routeru bude odpojen přívodní kabel internetu přibližně na 10 sekund. Jakmile bude kabel připojen, začne měření času, za který bude adaptér schopen obnovit spojení s cloudem. Tento případ bude simulovat výpadek spojení mezi domácím routerem a cloudem (například u lokálního poskytovatele internetu). Měření začne, jakmile bude přívodní kabel připojen k domácímu routeru. Měření bude ukončeno ve chvíli, kdy adaptér přijme první odpověď na příkaz ping od cloudu.

Každý tento pokus bude proveden 10× a časy budou zaznamenány. Výstupem tohoto pokusu bude graf, ve kterém budou jednotlivé fáze zobrazeny. Důležitým výstupem tohoto testu bude průměrná hodnota, která bude spočítána ze zaznamenaných 10 pokusů pro každou fázi.

Test tunelu na aplikační úrovni

Dalším testem OpenVPN tunelu bude jeho kontrola na aplikační úrovni. Předchozí testy jsou zaměřeny na síťovou vrstvu architektury TCP/IP. Tento test bude zaměřen na nejvyšší vrstvu, kterou je vrstva aplikační, pomocí jednoduchého dotazu na webový server a jeho odpověď. Webový server bude spuštěn pomocí služby netcat, která je ideální pro svou jednoduchost a rychlost použití. Webový server bude spuštěn na školním serveru ant-2. Z adaptéru bude zaslán dotaz na server, který po přijetí dotazu odpoví a s odpovědí zašle informační data (např. aktuální čas na serveru, vytížení HW, nebo nastavení síťového rozhraní). Tento test prověří, zda je možné skrze tunel zaslat příkaz z adaptéru na server. Na serveru dojde k jeho zpracování a odpovědi na tento příkaz.

Test izolovanosti klientů

Podle požadavků v kapitole 3.1 této práce bude otestováno, zda jsou klienti vzájemně izolovaní. To znamená, že se klienti nebudou moci navzájem ovlivňovat a ani jeden druhého nebudou schopni kontaktovat.

Budou spuštěni 2 OpenVPN klienti na různých zařízeních. Bude ověřeno, že oba klienti jsou připojeni k serveru a mohou s ním komunikovat (např. kontrolou log souboru, popř. příkazem ping). Následně bude izolovanost prověřena příkazem ping, který bude odeslán z jednoho zařízení na druhý a naopak. Pokud příkaz ping projde, nejsou klienti izolovaní.

4 Implementace

V této kapitole jsou ukázky jednotlivých konfiguračních souborů, které jsou na serveru a adaptéru. Je zde podrobně popsán každý z těchto souborů. V další části je kopie skriptu, který je použit pro testování. Dále je zde uveden seznam problémů, které při implementaci nastaly. Ke každému problému jsou sepsaná možná řešení a jedno řešení je v rámci této práce vybráno a použito.

4.1 Server

V této části práce je ukázka konfiguračního souboru běžícího na serveru. Tento konfigurační soubor je možné najít na serveru ve složce: „`/etc/openvpn/`“.

Na následujících řádcích je ukázka konfiguračního souboru s popisem jednotlivých příkazů:

```
dev tap
port 1194
proto tcp-server
mode server
```

Předchozí 4 řádky popisují, jaké virtuální síťové rozhraní bude použito. Je použit výchozí port pro OpenVPN a komunikuje se pomocí TCP protokolu. Poslední řádek naznačuje, že OpenVPN běží v módu `server`.

```
ifconfig 10.0.0.1 255.255.0.0
ifconfig-pool 10.0.0.2 10.0.255.254 255.255.0.0
```

Na těchto 2 řádcích je určeno, jakou IP adresu má server a jaké IP adresy jsou přidělovány klientům, kteří se k serveru připojují.

```
tls-server
dh /etc/openvpn/dh1024.pem
ca /etc/openvpn/ca.crt
cert /etc/openvpn/server.crt
key /etc/openvpn/server.key
```

V předchozích 5 řádcích jsou veškeré potřebné informace k zabezpečení OpenVPN síťového tunelu. Je zde popsáno, jaké je použito šifrování a cesty k potřebným klíčům a certifikační autoritě.

```
log-append /etc/openvpn/openvpn.log
```

Tento řádek určuje, kam jsou ukládány logy o běhu OpenVPN serveru.

```
user nobody
group nogroup
```

Zde je popsáno, kterému uživateli OpenVPN server patří a do které skupiny uživatelů náleží. Pokud by zde tyto příkazy nebyly, OpenVPN server by byl spouštěn s právy roota.

`comp-lzo`

Tady je informace o komprimaci přenášených dat. Pokud by data nebyla komprimována, byl by přenášený obsah veliký a tím by se zatěžovala linka. Komprimace je prováděna na úkor zátěže procesoru.

`keepalive 10 120`

Tímto příkazem se zaručuje, že spojení nebude přerušeno v důsledku neaktivity. Tento příkaz zařídí, že každých 10 sekund se vyšle ping, který udržuje tunel aktivní a pokud se do 120 sekund nevrátí žádný vyslaný ping, tak se server restartuje.

`ping-timer-rem`

Tato volba se používá u serveru, aby ignoroval volbu `keepalive` v případě, že není připojený žádný klient.

`persist-tun`

`persist-key`

Pokud se OpenVPN pokusí spojení restartovat, nemusí už mít potřebná oprávnění k vytvoření virtuálního síťového rozhraní či k opětovnému přečtení klíče. Z tohoto důvodu se používají předchozí dva příkazy. První z nich zajistí, že OpenVPN v případě restartu nezahodí virtuální síťové rozhraní, ale ponechá si jej. Druhý zajistí, že i po restartu se bude OpenVPN moci nadále číst potřebné klíče.

`verb 3`

Příkaz `verb` určuje, jak podrobné informace jsou zapisovány do log souboru. Jako parametr tohoto příkazu je možné zvolit hodnoty 0-9, kde 0 znamená žádné záznamy v log souboru a číslo 9 maximální „upovídanost“.

`duplicate-cn`

Tato volba povoluje současné přihlášení více klientů se stejným certifikátem. Tato volba je v našem případě používána především pro testování, v případném komerčním použití by tato volba byla odstraněna.

`mute 10`

Posledním příkazem serverového konfiguračního souboru je `mute`. Jeho parametr určuje, kolik po sobě jdoucích stejných řádků v log souboru je nahrazeno jedním a zbylé budou smazány. Díky tomuto příkazu nevznikají duplicitní řádky a díky tomu je log soubor lépe čitelný.

Ukázka konfiguračního souboru je také v příloze. (viz. Příloha č. 2)

4.2 Klient

V této podkapitole je ukázka konfiguračního souboru, který je uložený na adaptéru. Tento soubor je na adaptéru dostupný ve složce: „`/etc/openvpn/`“.
Toto je ukázka konfiguračního souboru s popisem jednotlivých příkazů:

```
remote ant-2.fit.vutbr.cz
```

Tímto příkazem klientu řekneme, k jakému serveru se klient připojuje.

```
client
```

Pomocí příkazu `client` je při spuštění OpenVPN vytvořený proces pro klienta.

```
port 1194
proto tcp
dev tap
```

Tyto volby jsou stejné jako u konfiguračního souboru serveru. Nastavuje se pomocí nich port, protokol a virtuální síťové rozhranní.

```
log-append /etc/openvpn/openvpn.log
```

Tento řádek určuje, kam jsou ukládány logy o běhu OpenVPN klienta.

```
tls-client
ca /etc/openvpn/ca.crt
cert /etc/openvpn/client.crt
key /etc/openvpn/client.key
```

Podobně jako u serveru se těmito příkazy nastavuje způsob šifrování dat a umístění potřebných klíčů a certifikátu.

```
keepalive 10 120
ping-timer-rem
mute 10
comp-lzo
verb 3
```

Předcházející příkazy jsou shodné s příkazy v kapitole 4.1 této práce.

Ukázka konfiguračního souboru je také v příloze. (viz. Příloha č. 3)

Testovací skript

Na následujících řádcích je ukázka skriptu, který je vytvořen za účelem testování. Tento skript se zabývá tvorbou síťových komunikačních tunelů pomocí OpenVPN. Pro vytvoření více tunelů stačí mít ve složce odpovídající OpenVPN vytvořeno požadované množství konfiguračních souborů. Tento skript je dostupný jako Příloha č. 4. Spuštění tohoto skriptu je možné ve složce „`/etc/openvpn/`“ příkazem „`sh script.sh`“.

Na prvním řádku musí být označení, že se jedná o skript vytvořený v jazyce bash, aby bylo jasné, jak s ním zacházet.

```
#!/bin/bash
echo -n Zadejte pocet klientu, ktere chcete vytvorit:
read pocet
```

Na těchto dvou řádcích je od uživatele vyžadováno číslo, které určí kolik OpenVPN klientů bude vytvořeno.

```
for i in `seq 0 $pocet`;
do
cp vpn_client vpn_client$i\.conf;
echo "log-append /etc/openvpn/openvpn"$i".log" >> vpn_client$i\.conf
done
```

V tomto cyklu se inkrementuje hodnota `i` až po hodnotu, kterou zadal uživatel v předchozím kroku. V každém průchodu cyklu se zkopíruje výchozí konfigurační soubor, kterému se přidá hodnota `i` a koncovka `.conf`. Výchozí konfigurační soubor je bez přípony, aby nebyl spouštěn jako VPN klient při spuštění zařízení. Na dalším řádku se do nově vytvořeného konfiguračního souboru přidá řádek, který určuje, jaký a kde bude vytvořen soubor, kam se budou ukládat logy.

```
while :;
do
        if test -e /etc/openvpn/vpn_client$pocet\.conf
        then
                rm vpn_client$pocet\.conf
                rm openvpn$pocet\.log
        else
                break
        fi
        pocet=$(expr $pocet + 1)
done
```

V tomto nekonečném cyklu se mažou konfigurační soubory a log soubory s pořadovým číslem větším než číslem, které bylo zadáno na začátku skriptu. Postupně se v každém průchodu tohoto cyklu zvyšuje hodnota `pocet` až do doby, kdy existují ještě nějaké konfigurační soubory s touto hodnotou. Jakmile neexistuje žádný takový soubor, tak je tento cyklus ukončen. Posledním příkazem se restartuje OpenVPN a díky tomu se spustí požadované množství klientů.

```
/etc/init.d/openvpn restart;
```

4.3 Problémy při implementaci

Při implementaci OpenVPN na adaptér nastaly některé problémy. V této kapitole jsou popsány jejich projevy, příčiny a možná řešení (případně aplikovaná řešení).

Datum

Při spuštění adaptéru je nastaveno výchozí datum a to konkrétně: 1. 1. 1970. Jelikož adaptér nemá vestavěnu baterii, nedokáže si udržovat informaci o datu a čase, proto se při zapnutí adaptéru čas a datum nastaví na výchozí hodnotu. Problém nastává v případě kontroly certifikátů na jejich platnost. Jelikož jsou certifikáty vystaveny s novějším datem než rok 1970, vypíše OpenVPN chybovou hlášku: „*certificate is not yet valid*“, což v překladu znamená, že certifikát ještě není platný. A díky tomuto problému není možné navázat síťový tunel.

Možností, jak vyřešit tento problém, je několik:

1. Vytvoření certifikátů s datem starším než 1970. Tímto způsobem je možné rychle a snadno tento problém vyřešit, ale toto řešení není příliš kvalitní. V případě vytvoření nové certifikační autority s tímto může být znovu problém.
2. Nainstalování softwaru pro aktualizaci data pomocí internetového připojení. Tento SW se připojí na vzdálený server, z kterého je možné získat aktuální čas (ve většině případů od atomových hodin). Problém tohoto řešení je, že není příliš časově efektivní. Tím, jakou rychlostí se nastaví hodiny, se zabývám v kapitole testování.
3. Další možností je vytvořit jednoduchý skript, který při každém spuštění adaptéru aktualizuje datum na aktuálnější hodnotu. Díky tomu bude v době spuštění OpenVPN datum vyhovovat kontrole platnosti certifikátů.
4. Dále je možné kombinovat předchozí dvě varianty. Díky tomu se při startu adaptéru nastaví datum, které vyhovuje kontrole platnosti certifikátů. A navíc je po nějaké době datum upraveno na aktuální hodnotu, pokud má adaptér přístup na internet. Toto řešení je zvoleno i v našem případě, protože je shledáno jako nejefektivnější.

Na následujících řádcích je jednoduchý skript, který je uložen v „`/etc/init.d/`“. Skript je nastaven tak, aby se při spuštění systému vykonal.

Skript nastaví hodnotu date na datum: 22. 4. 2014 00:00. Na začátku skriptu je informace o tom, že je to skript napsaný v jazyce bash. Při spuštění systému se tento skript zavolá s parametrem `start`, díky kterému se vykoná příkaz `date` a nastaví se datum.

```
#!/bin/sh
case "$1" in
start)
      date 042200002014
```

```
;;
*)
esac;
```

Možným rozšířením tohoto skriptu je ukládat aktuální datum, které se aktualizuje jako správné, pomocí nainstalované služby. Díky tomu by bylo zaručeno, že v případě restartování adaptéru se načte toto datum.

Zabíjení procesu

Při spuštění OpenVPN z terminálu přes ssh se proces OpenVPN naváže na spuštěný terminál a při jeho zavření se proces ukončí. Díky tomu při každém uzavření terminálu skončí i OpenVPN tunel. Zmiňovaný problém nastával i při spuštění procesu na pozadí. Řešením problému je použití sériové linky pomocí UART kabelu, který propojí adaptér a PC. Na jedné straně kabelu je USB konektor a na druhé jsou 3 piny, které je zapotřebí správně připojit k adaptéru. Dále je zapotřebí do PC nainstalovat ovladač, který umožní USB portu chovat se jako sériový port.

Pokud by chtěl uživatel spouštět OpenVPN z terminálu pomocí ssh tunelu, musí mít OpenVPN nastaveno, aby se spustilo při bootování systému adaptéru a při každé změně v nastavení OpenVPN by se musel adaptér restartovat. Pak by tento proces nebyl navázán na spuštěný terminál a při jeho vypnutí by se tento proces neukončil.

5 Testování

Tato kapitola se zaměřuje na testování navrženého OpenVPN tunelu. Testování je zaměřeno na požadavky z kapitoly 3.1 této práce. Cílem testování je ověřit, zda OpenVPN tunel odpovídá zadaným požadavkům a je tak možné ho použít pro realizace tunelu mezi adaptérem a serverem. V případě, že by tunel nevyhovoval některému ze specifikovaných požadavků, nebylo by možné ho ve finální podobě inteligentní domácnosti použít a muselo by se najít jiné řešení tohoto problému.

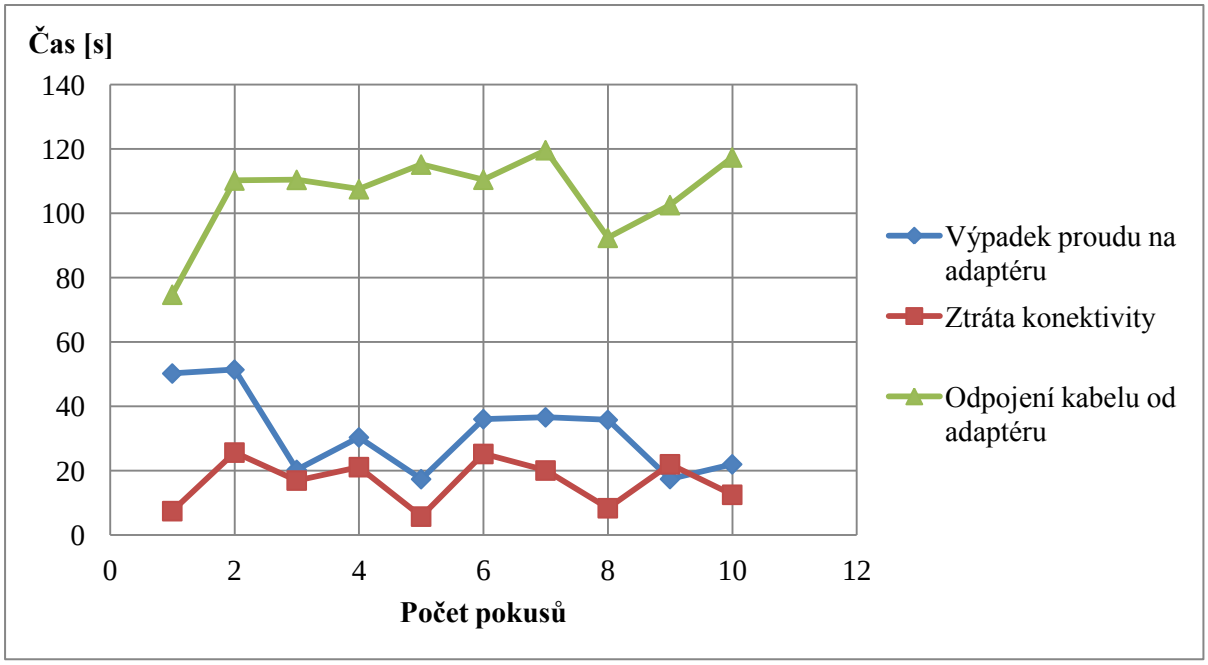
5.1 Reálné situace v domácnosti

Tato část testování je zaměřena na situace, které v reálné domácnosti mohou nastat. Jak je navrženo v kapitole č. 3 této práce, tak probíhalo testování. V obrázku č. 4 jsou zaznamenány časy z jednotlivých experimentů.

Časy, které byly naměřeny při odpojení síťového kabelu od adaptéru, jsou v grafu vyneseny zelenou spojnicí. Tato situace simulovala výpadek domácího routeru. Čas byl měřen od chvíle, kdy byl kabel znovu připojen k adaptéru, až do chvíle kdy byla přijata odpověď na příkaz ping od cloudu. Tento způsob měření byl zvolen z důvodu eliminace doby restartování domácího routeru. Jelikož každý domácí router startuje jinak dlouhou dobu, tímto způsobem byl tento efekt odstraněn. V grafu je vidět 10 naměřených hodnot a v průměru trvala obnova spojení mezi adaptérem a serverem 106 s. Pokud k této době připočítáme i dobu restartování domácího routeru získáme poměrně dlouhý čas, kdy adaptér není schopný komunikovat se serverem. V další kapitole je rozebráno možné řešení.

Modrá spojnice představuje časy, které byly naměřeny při odpojení napájení adaptéru. Tento experiment simuluje výpadek proudu, který napájí adaptér, a simuluje reakci na restartování adaptéru. Doba od připojení napájení k adaptéru byla měřena až do doby, kdy byla na cloudu přijata odpověď na příkaz ping. Průměrně trvalo adaptéru nastartování a obnovení spojení se serverem 31,7 s. To jakým způsobem eliminovat tento čas se zabývá další kapitola.

Červená spojnice zobrazuje časy, které byly naměřeny po připojení přívodního kabelu do routeru. Tento pokus měl za úkol simulovat, jak dlouho bude trvat obnova spojení mezi adaptérem a cloudem v případě, že na cestě mezi adaptérem a cloudem nebude možné spojení. Tento test simuloval výpadek například u lokálního poskytovatele internetu. Na adaptéru je spuštěn příkaz ping a po připojení přívodního kabelu do domácího routeru je měřen čas, za jak dlouho se obnoví spojení mezi adaptérem a cloudem. V průměru to trvalo 16,5 s. Tento efekt téměř není možné eliminovat a proto je potřeba počítat s tímto problémem při navrhování inteligentní domácnosti.



Obrázek 4: Testování adaptéru na reálné situace, které v domácnosti mohou nastat.

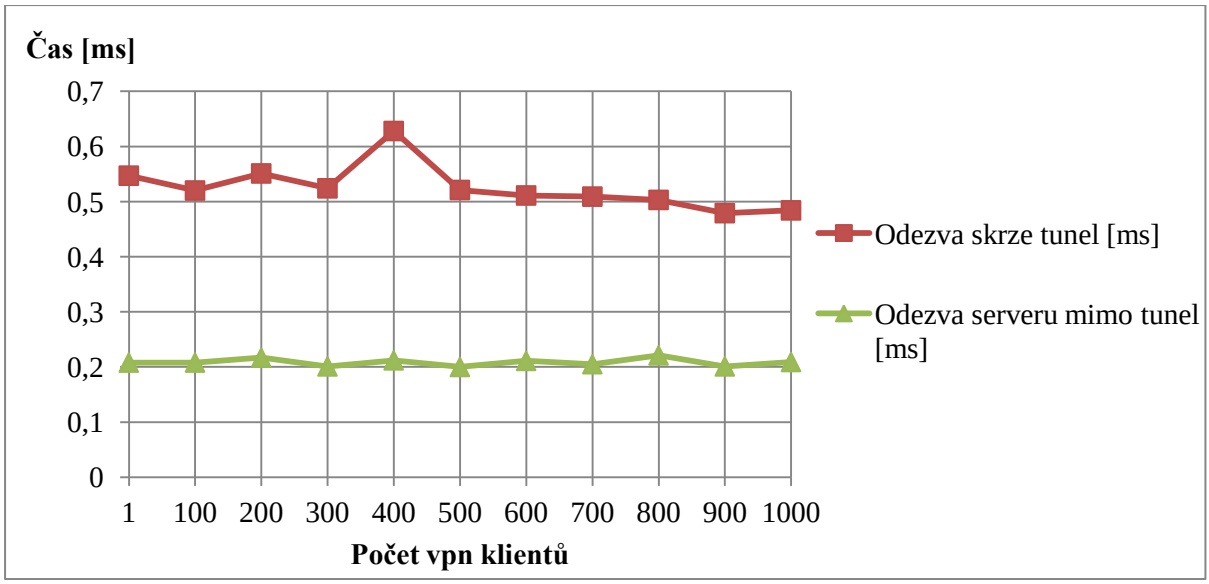
Toto testování prokázalo, že časy obnovení spojení mezi adaptérem a cloudem nejsou zanedbatelné a je zapotřebí omezit možnost vzniku těchto situací. Záložní zdroj by výrazně omezil tuto prodlevu v případě zelené a modré spojnice (tj. výpadek proudu na adaptéru a odpojení kabelu od adaptéru). Pokud by byl adaptér i router připojen na záložní zdroj, jediná prodleva, která by vznikla, je prodleva v podobě červené spojnice (ztráta konektivity) a díky tomu by byla maximální prodleva okolo 16 s. Prodlevu při ztrátě konektivity nemůžeme nijak ovlivnit a proto je potřeba s ní počítat.

5.2 Testování odezvy cloudu na počet připojených vpn klientů

Toto testování bylo zaměřeno na odezvu cloudu v případě, že se bude počet připojených klientů zvyšovat. Testování probíhalo v rámci školní sítě, proto jsou naměřené časy v řádu desetin milisekund. Testování bylo plánované provádět mimo školní síť, ale při spuštění 400 a více vpn klientů lokální síť selhávala a měření bylo zkreslené. Školní síť si s tímto problémem snadno poradila.

V obrázku č. 5 je vidět, že při použití vpn tunelu se čas více než zdvojnásobil. Tento nárůst by byl při finálním použití neakceptovatelný. Tento nárůst je způsobený tím, že pakety putují školní sítí velkou rychlostí a tak zde největší čas zabírá zabalení paketu na straně klienta a rozbalení na straně serveru. Při použití v běžné síti je tento nárůst okolo 25 %. Průměrná hodnota při poslání 20 ping dotazů změřena z domácí sítě (okres Kolín) při spuštění jednoho vpn klienta byla 14,47 ms. Při vypnutí vpn klienta a při stejném měření byla hodnota odezvy 11,64 ms. Při stejném měření v Brně, ale mimo školní síť, byl čas mimo vpn tunel 15,23 ms a skrze vpn tunel bylo naměřeno 19,53 ms.

Dále je v grafu vidět, že se zvyšujícím se počtem vpn klientů se časy nezvyšují. To znamená, že při připojení 1000 klientů se odpovědi serveru neprodlužují a dá se odhadnout, že cloud s tímto HW zvládne obsloužit mnohem více vpn klientů. Jediná odchylka vznikla při odezvě skrze tunel při 400 připojených klientech, ale tato odchylka je zhruba 0,1 ms, což je zanedbatelná hodnota, která mohla vzniknout momentálním vytížením sítě.



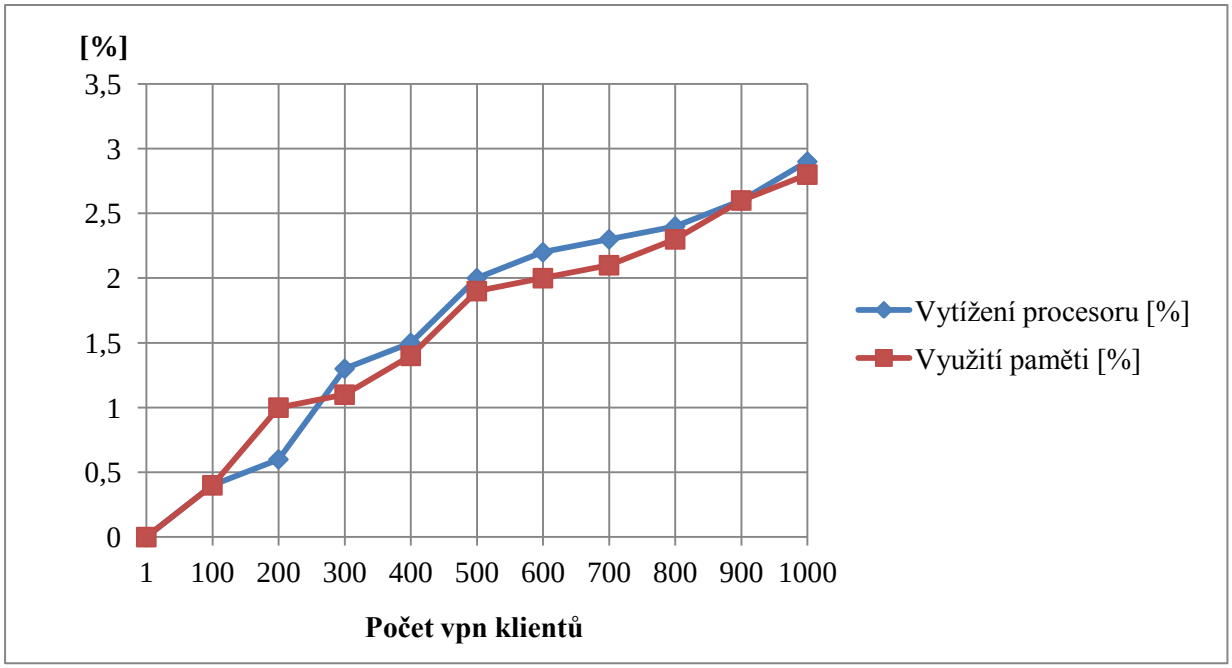
Obrázek 5: Testování odezvy cloudu na počet připojených klientů.

5.3 Testování nároků na HW cloudu

Toto testování bylo zaměřeno na vytížení HW. Testování probíhalo na serveru ant-2 a jeho HW parametry jsou popsány v kapitole 3.4 této práce.

V obrázku č. 6 je vidět, že vytížení procesoru a využití pamětí je přibližně stejné v závislosti na připojených vpn klientech. Se zvyšujícím se počtem připojených klientů téměř lineárně rostly hodnoty vytížení procesoru a využití pamětí.

Díky tomuto testu je možné odhadnout maximální množství vpn klientů připojených na jeden server (konkrétně ant-2). Jelikož při 1000 připojených vpn klientech je vytížení procesoru a využití paměti zhruba 3%, tak maximální počet vpn klientů je více než 33 000. Toto číslo je ale v praxi nereálné, protože tento server nebude obsluhovat pouze vpn klienty, ale bude mít na starosti komplexní obsluhu inteligentní domácnosti. Dále tato data nejsou ovlivněna provozem mezi vpn klientem a serverem. Jediné, čím tito vpn klienti server zatěžovali, byly pakety ping of live, které posílá každý vpn klient, aby udržel komunikační tunel otevřený.



Obrázek 6: Nároky na HW cloudu.

5.4 Test průchodnosti tunelu na aplikační vrstvě

Tento test by podle návrhu z kapitoly 3.4 této práce měl prověřit, zda je možné skrze tunel zaslat příkaz z adaptéru na server. Na serveru tento dotaz zpracovat a poslat na něj odpověď zpět na adaptér.

Pro implementaci tohoto testu byla zvolena služba netcat, která je pro svou jednoduchost implementace a použití ideální. Na serveru byl spuštěn jednoduchý webový server, který vysílá na portu 10000. Webový server je spuštěn na adrese localhost.

```
while true; do { echo -e 'HTTP/1.0 200 OK\r\n'; sh test; } | nc -l
10000; done
```

Tento jednoduchý příkaz je spuštěn na straně serveru. Na port 10000 a adresu localhost jsou pomocí tohoto skriptu přijímány příkazy. Pokud přijde dotaz na webový server, tak na něj odpoví `HTTP/1.0 200 OK\r\n` a spustí skript `test`, který zašle informace o serveru. Konkrétně zasílá informace HW serveru a informace o síťových rozhraních (příkaz `ifconfig`). Obsah souboru `test` je v příloze č. 4.

```
echo -n "GET / HTTP/1.0\r\n\r\n" | nc -n 10.0.0.1 10000 -vv
```

Tento příkaz je spuštěn na adaptéru. Výstup příkazu echo je poslán skrze rouru programu netcat. Program netcat vyšle příkaz na adresu 10.0.0.1 a port 10000 a čeká na odpověď. Jakmile přijde odpověď, vypíše její obsah na obrazovku. Na obrazovku se vypíší informace, které zasílal

server. Konkrétně informace o HW serveru a informace o síťových rozhraních. Díky tomu, že byla vypsána odpověď od serveru, byla ověřena průchodnost síťového komunikačního tunelu na aplikační vrstvě.

5.5 Test izolovanosti klientů

Byli spuštěni 2 OpenVPN klienti, každý na jiném zařízení. Příkazem ping bylo ověřeno správné fungování OpenVPN tunelu vůči serveru. U obou klientů příkaz ping proběhl správně a díky tomu je zaručena správná funkčnost tunelu. Pro ověření, že jsou klienti vzájemně izolováni, byl spuštěn příkaz ping z jednoho klienta na druhý a naopak. V obou případech neprošel žádný paket a díky tomu se dá určit, že jsou klienti izolováni a nemohou spolu vzájemně komunikovat ani se jinak ovlivňovat skrze OpenVPN tunel.

5.6 Zhodnocení testování

Testováním bylo ověřeno, že OpenVPN splňuje veškeré požadavky (viz. kapitola 3.4 této práce), které jsou zapotřebí pro síťový komunikační tunel mezi adaptérem a cloudem v rámci inteligentní domácnosti. V případě reakce na ztrátu síťového spojení je zapotřebí vnějších zdrojů pro odstranění prodlevy při obnovení komunikace mezi adaptérem a serverem. Tímto vnějším zdrojem je záložní zdroj energie, kterým bude zajištěn stálý přísun elektrické energie adaptéru a domácímu routeru i v případě výpadku proudu v domácnosti. Díky tomuto zdroji nenastane výpadek adaptéru ani domácího routeru, což odbourá zpoždění při jejich startování a obnovování spojení. Jediný výpadek, který neovlivníme, je výpadek konektivity mimo inteligentní domácnost. Omezení tohoto výpadku zajistí pouze spolehlivý poskytovatel internetu.

6 Závěr

Cílem této práce bylo navrhnout, implementovat a otestovat síťový komunikační tunel v rámci inteligentní domácnosti. Tento síťový komunikační tunel spojuje adaptér a cloud. Hlavními požadavky tohoto komunikačního tunelu byly bezpečnost přenášených dat a jejich spolehlivé doručení s co nejmenším zpožděním oproti běžnému přenosu skrze internet.

V této práci je zkoumáno několik možností, jak tento komunikační tunel řešit. Podle teoretického rozboru jednotlivých řešení je zvoleno řešení pomocí OpenVPN, které nejlépe splňovalo zadané požadavky. Při implementaci se projevilo několik problémů, které musely být operativně vyřešeny. Problém a jejich řešení jsou v této práci také popsány.

Důležitým faktorem pro to, jestli je OpenVPN vhodné pro konečné použití v inteligentní domácnosti, je splnění všech předem definovaných požadavků. Na splnění těchto požadavků bylo navrženo několik testů. Tyto testy byly následně vykonány a podle výsledků je zřejmé, že OpenVPN splňuje všechny zadané požadavky. Díky tomu je OpenVPN vhodným řešením pro vytvoření komunikačního tunelu mezi adaptérem a cloudem v inteligentní domácnosti. Dále je v této práci navrženo několik vylepšení, které pomohou k ještě lepším výsledkům této implementace v rámci inteligentní domácnosti.

Hlavním přínosem této práce je odhalení nedostatků, které nastávají při implementaci OpenVPN na naší vývojové platformě. Zásadním problémem naší platformy je chybějící baterie, která by udržovala informace o čase a datu v době, kdy je adaptér odpojen od elektrické energie.

Literatura

[1] Elektřina a rozvody. *DigiRoom.cz* [online]. [cit. 2014-02-03]. Dostupné z: http://digiroom.digizone.cz/special/elektrina-a-rozvody/

[2] Topení a klimatizace. *DigiRoom.cz* [online]. [cit. 2014-02-03]. Dostupné z: http://digiroom.digizone.cz/special/topeni-a-klimatizace/

[3] Umělé a přírodní osvětlení. *DigiRoom.cz* [online]. [cit. 2014-02-03]. Dostupné z: http://digiroom.digizone.cz/osvetleni-prirodni-svetlo/

[4] Bezpečnost a dohled. *DigiRoom.cz* [online]. [cit. 2014-02-03]. Dostupné z: http://digiroom.digizone.cz/special/bezpecnost-a-dohled/

[5] Smarthome architecture. *IOT Wiki* 2014 [cit. 2014-05-19]. Dostupné neveřejně z: https://merlin.fit.vutbr.cz/wiki-iot/index.php/Smarthome_architecture

[6] Wikipedia contributors. MiWi. *Wikipedia, The Free Encyclopedia*. 2013 [cit. 2014-01-24]. Dostupné z: http://en.wikipedia.org/wiki/MiWi

[7] THOMAS, Thomas M. *Zabezpečení počítačových sítí bez předchozích znalostí*. Vyd. 1. Brno: CP Books, 2005. ISBN 80-251-0417-6.

[8] *Bezpečnost sítí: Velká kniha*. Vyd. 1. Brno: CP Books, 2005, 589 s. ISBN 80-251-0697-7.

[9] Wikipedia contributors. Virtual Private Network. *Wikipedia, The Free Encyclopedia.* 2014 [cit. 2014-01-24]. Dostupné z: http://de.wikipedia.org/wiki/Virtual_Private_Network

[10] PUŽMANOVÁ, Rita. *TCP/IP v kostce*. 1. vyd. České Budějovice: Kopp, 2004, 607 s. ISBN 80-723-2236-2.

[11] Security Architecture for the Internet Protocol. *RFC 2401* [online]. [cit. 2014-02-03]. Dostupné z: http://www.ietf.org/rfc/rfc2401.txt

[12] IP security. *RFC 2411* [online]. [cit. 2014-02-03]. Dostupné z: http://www.ietf.org/rfc/rfc2411.txt

[13] HLADÍK, Radek. OpenVPN - VPN jednoduše. *Root.cz* [online]. 2004 [cit. 2014-02-03]. Dostupné z: http://www.root.cz/clanky/openvpn-vpn-jednoduse/

Seznam příloh

Příloha 1: CD

Obsah CD:

- Bakalářská práce.pdf
- vpn_client.conf
- vpn_server.conf
- script.sh
- test

Příloha 2: Konfigurační soubor serveru

```
dev tap
port 1194
proto tcp-server
mode server
ifconfig 10.0.0.1 255.255.0.0
ifconfig-pool 10.0.0.2 10.0.255.254 255.255.0.0
tls-server
dh /etc/openvpn/dh1024.pem
ca /etc/openvpn/ca.crt
cert /etc/openvpn/server.crt
key /etc/openvpn/server.key

log-append /etc/openvpn/openvpn.log
user nobody
group nogroup
comp-lzo
keepalive 10 120
ping-timer-rem
persist-tun
persist-key
mute 10
verb  6
duplicate-cn
```

Příloha 3: Konfigurační soubor klienta

```
remote ant-2.fit.vutbr.cz
client
resolv-retry infinite
tls-client
port 1194
proto tcp
dev tap
log-append /etc/openvpn/openvpn.log
ca /etc/openvpn/ca.crt
cert /etc/openvpn/client.crt
key /etc/openvpn/client.key
keepalive 10 120
ping-timer-rem
persist-tun
persist-key
mute 10
comp-lzo
verb 3
```

Příloha 4: Testovací skript

```
#!/bin/bash

#v adresari musi existovat soubor vpn_client
#vyzadani poctu klientu
echo -n Zadejte pocet clientu ktere chcete vytvorit:
read pocet
#tady se nakopiruje pozadovany pocet konfiguracnich souboru
for i in `seq 1 $pocet`;
do
     cp vpn_client.conf vpn_client$i\.conf;
     echo     "log-append     /etc/openvpn/openvpn"$i".log"     >>
vpn_client$i\.conf
done
#timto cyklem se uklidi konfiguracni soubory co tam nemaji byt
while :;
do

     if test -e /etc/openvpn/vpn_client$pocet\.conf
     then
          rm vpn_client$pocet\.conf
          rm openvpn$pocet\.log
     else
          break
     fi
     pocet=$(expr $pocet + 1)
done
#restart openvpn (zastavi se vsechny bezici klienti a spusti se
pozadovany
#pocet klientu
/etc/init.d/openvpn restart;
```

Příloha 5: Soubor `test`

```
#!/bin/bash
echo "************PRINT SOME TEXT***************\n"
echo "Hello World!!!"
echo "\n"

echo "Resources:"
vmstat -S M
echo "\n"

echo "Addresses:"
echo "$(ifconfig)"
echo "\n"
```